数式処理システムにおけるMiddle Regulatorの開発 (数式処理における理論と応用の研究)

AUTHOR(S):

出口, 博章

CITATION:

出口, 博章. 数式処理システムにおけるMiddle Regulatorの開発 (数式処理における理論と応用の研究). 数理解析研究所講究録 1999, 1085: 1-8

ISSUE DATE:

1999-03

URL:

http://hdl.handle.net/2433/62816

数理解析研究所講究録
1085巻 1999年 1-8

数式処理システムにおける Middle Regulator の開発

神戸大学大学院自然科学研究科
出口博章 (Hiroaki DEGUCHI) *

1 はじめに

WWW（World Widw Web）における情報の閲覧の一画面のことはホームページ、またはウェブページと呼ばれるが、本稿ではウェブページという名称を使用する。

ミドルレギュレータ（Middle-regulator）の概念に関しては文献 [1] に詳しい説明がある。本稿では簡単に概要を述べる。

2 ミドルレギュレータの作成

ミドルレギュレータとは、ウェブブラウザをユーザインターフェースとして数式処理システムを利用するために、ブラウザと数式処理システムの間に位置するものである (図 1)。数式処理システムをウェブページのオブジェクトとして組み込むためのインターフェースとしての機能と、異なる種類の数式処理システムを同じ様な操作で利用するために、数式処理システム独自の表記法を変換する翻訳機能の二つの機能を持つ。今回作成したものはブラウザ側へのインターフェース部を Perl で CG I(Common Gateway Interface）プログラムとして作成し、数式処理システム側へのインターフェース部は Perl あるいは C で作成しており、その二つの部分をあわせてミドルレギュレータと呼ぶことにする (図 2)。

数式処理システムがフロントエンドとカーネルによって構成されており、その間で行なわれている通信の方法が公開されている場合は、ミドルレギュレータはその数式処理システムのカーネルに接続して利用する (図 3)。Wolfram Research, Inc. の Mathematica を使用する場合は、MathLink([2][3]) を利用することによってカーネルに接続し、カーネルの数式処理機能を利用できるため MathLink によってカーネルを直接利用する。

また、実際の演算を行なう計算機は UNIX 系のシステムをサーバとして利用することを前提としている。よって、数式処理システムがフロントエンドとカーネルに分かれていな

*deg@kobe-u.ac.jp

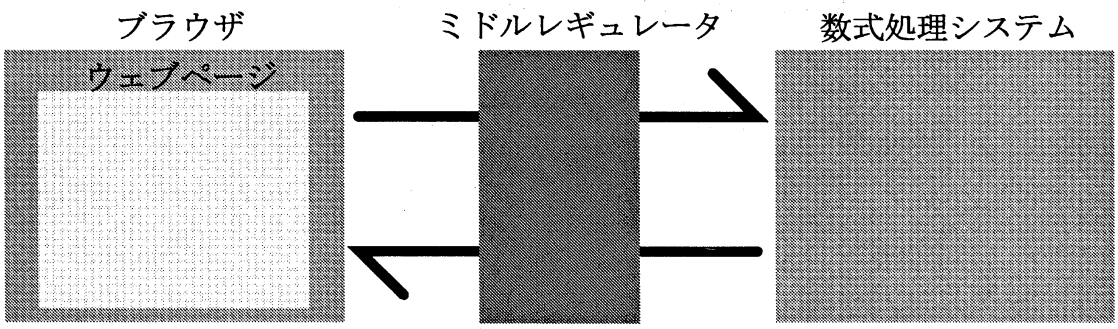

ブラウザ
ウェブページ
ミドルレギュレータ
数式処理システム

図 1:

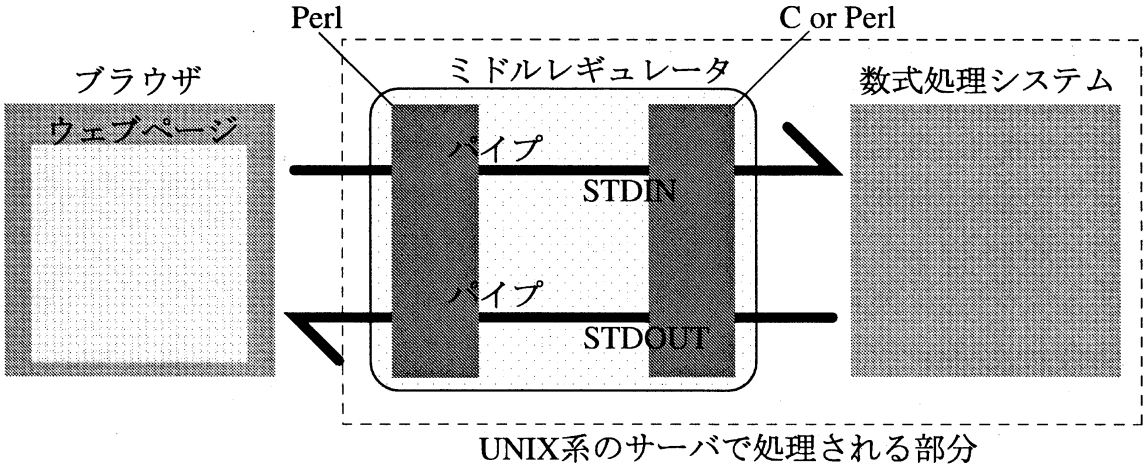

Perl
C or Perl
ブラウザ
ウェブページ
ミドルレギュレータ
数式処理システム
パイプ
STDIN
パイプ
STDOUT
UNIX系のサーバで処理される部分

図 2:

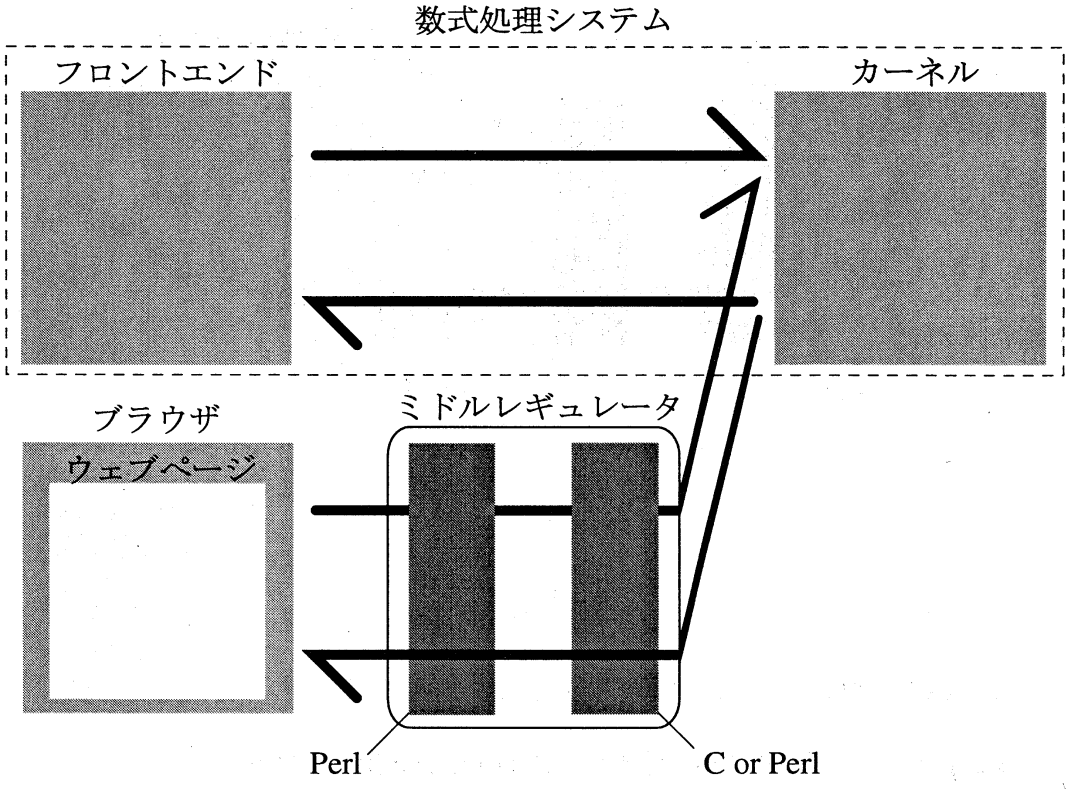


図 3:

い場合、あるいは分かれていてもその間での通信手段が公開されていない場合は、UNIXの標準入出力を利用してミドルレギュレータからパイプを通して数式処理システムを利用することになる (図 4)。富士通 HPC センターにおいて開発されている Risa/Asir([4]) の利用方法は今のところはミドルレギュレータで作成したパイプと Risa/Asir 側の標準入出力との接続での利用となる。

3 翻訳機能

正規表現を利用することによって、ミドルレギュレータはカーネルの差異を吸収することが可能になるが、ここではユーザが GUI を使わずに直接ミドルレギュレータにコマンドを送る場合を考える。カーネルは Mathematica の場合を例にとる。

`Expand[eq]` は eq を展開する関数であるが「eq を展開」と表現できたほうが、日本語を母国語とするユーザにとっては覚えやすい。同様に、`Plus[A,B]` は A と B を加えたものを計算する関数であるが、「A 足す B」や「A と B を足す」と命令出来たほうがユーザには覚えやすい。

このような変換を行なうために、変数`$input` にユーザからの入力が代入されているも

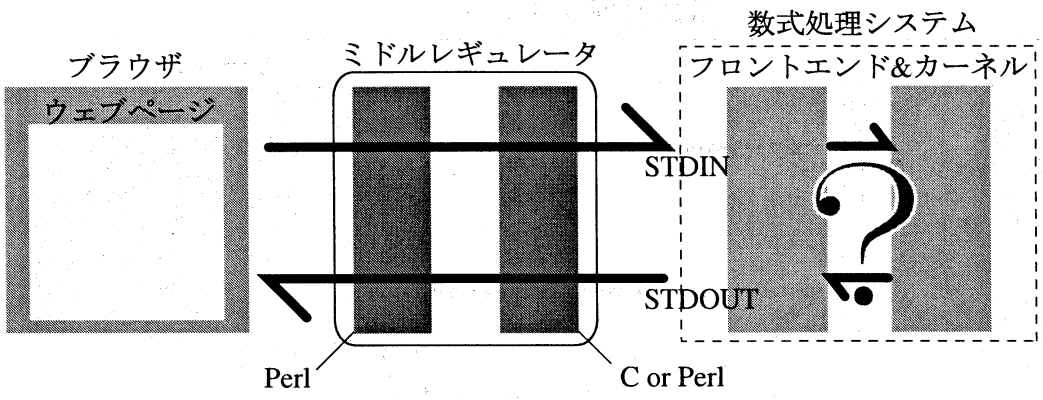


図 4:

のとすると、Perl では以下のように書くことによって実現できる。

```
$input =~ s/(\d+)\s*足す\s*(\d+)/Plus[$1,$2]/g;
$input =~ s/(\d+)\s*と\s*(\d+)\s*を足す/Plus[$1,$2]/g;
$input =~ s/(.*) を展開/Expand[$1]/g;
```

「$input =~ s/マッチする条件/置換文字列/g;」のかたちをしており、置換文字列中の$1や$2などはマッチする条件のなかの「()」で囲まれた部分を前から順番に$1,$2,$3,$4,...としたものである。注意点は、マッチする条件どうしの衝突がないようにしておくことである。たとえば、上の例に付け加えて「展開」という単語を「Expand」に置換するようにも設定した場合、「eq を展開」を「Expand[eq]」に置換する前に「eq を Expand」と置換される可能性もあり、結果が予想と食い違う場合もありうる。

また、置換のための上記のような行が多くなると、ミドルレギュレータのソース内部ではなく外部にテーブルとして作成したほうが置換内容を編集するのに便利なため、パターンマッチの条件と置換文字列の対応テーブルを外部に作成して格納している (図 5)。データベース形式は UNIX で標準的に利用されている DBM 形式を利用している。

4 グラフィックス出力

Mathematica のグラフィックス出力は大域変数

$DisplayFunction

の値がデフォルトとして使用されるようになっているため、その値を

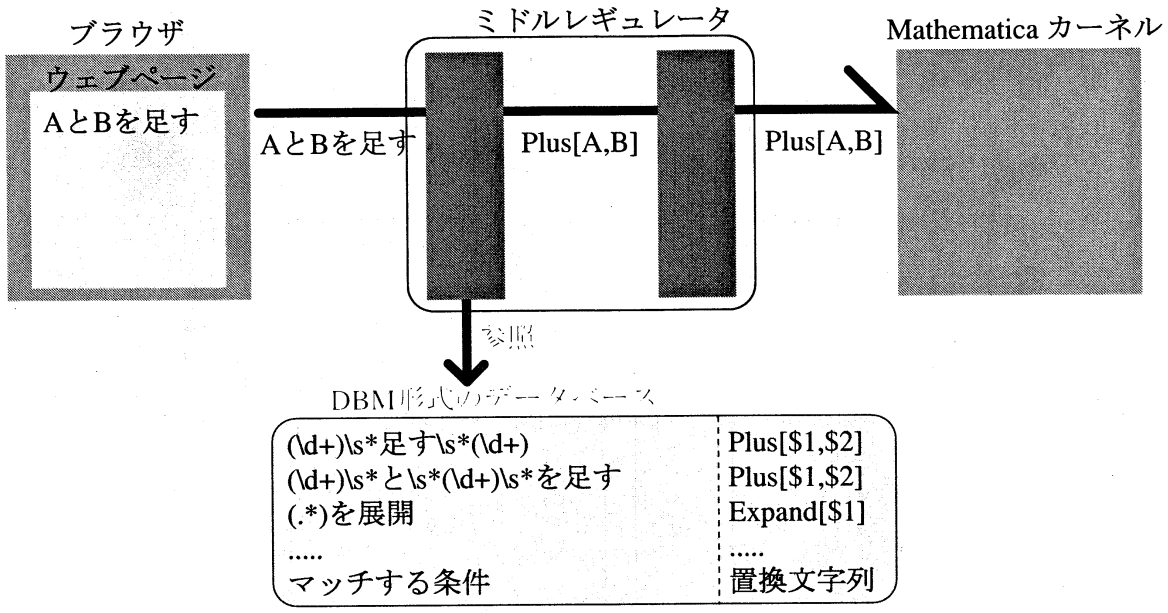


図 5: データの流れ

```
Display[$Display, \#]&
```

から

```
Display["/パス名/ファイル名", \#, "GIF"]&
```

に変更する。第三の引数は Ver.3 から利用できるようになったもので、出力グラフィックスの種類を指定する。このように`$DisplayFunction`の変更を最初に実行しておくことによって、それ以降はカーネルはグラフィックス出力を「/パス名/ファイル名」に GIF 形式で出力する。GIF 形式を選択した理由は、ウェブページにおける画像の標準的な形式の一つだからである。

ただし、このままで使用すると一つ問題点がある。グラフィックスを出力するファイル名を途中で変更するためには`$DisplayFunction`の値を変更するか、デフォルトのように`$DisplayFunction`内で`$Display`を使用しておいて`$Display`を変更する必要が出てくる。この状態ではグラフィックス出力を 2 回以上繰り返すと、前の出力結果は後の出力結果で上書きされることになり、複数のグラフィックス出力を同時にウェブページに表示することができない。

そこで、C プログラムの内部で、カーネルから MathLink のパケットタイプ「`RETURNTEXTPKT`」のパケットを拾った時にその文字列が「`-Graphics-`」か「`-SurfaceGraphics-`」に一致するかどうかを調べて、一致すれば、`$Display Function`で設定した「/パス名/ファイル名」のファイルをカウントアップしながら番号を付加した名前に変更し (図 6)、「`-Graphics-`」は「`-HTML`カウント番号`-\n-Graphics-`」、「`-SurfaceGraphics-`」は「`-HTML`カウント番

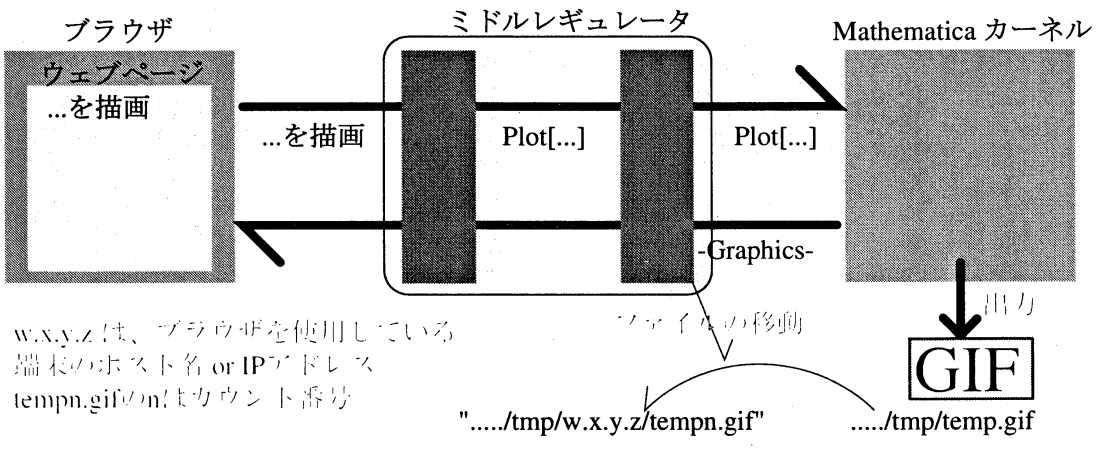


図 6: GIF ファイルの移動

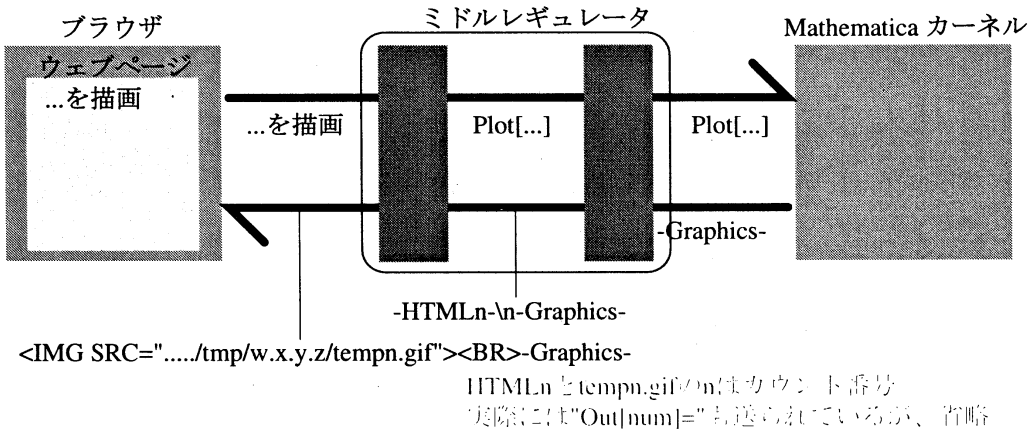


図 7: データの流れ

号-\n-SurfaceGraphics-」と変更して出力するようにして、P erl 部分で「-HTML カウント番号-」という文字列をイメージタグに書き換えてからウェブページとして出力すれば、そこにグラフィックスがリンクされて表示されることになる (図 7)。

この方法で、関数の後に「;」をつけた場合は「-Graphics-」や「-SurfaceGraphics-」がカーネルから返ってこないため、グラフィックスがウェブページに表示されない、という点のみが Mathematica のノートブックとの違いである (図 8)。

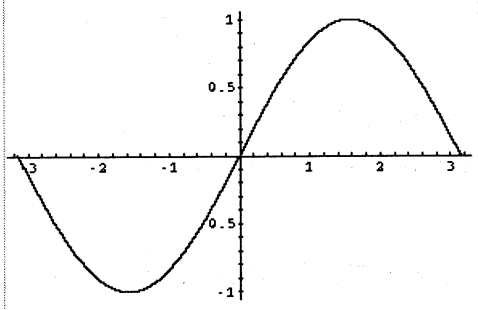
5 今後の課題

以上のようにミドルレギュレータを作成することによって、インターネット上でウェブブラウザを使用して数式処理システムを利用することを実現した。この方法の利点はイン

Mathematica出力

In[2]:= Plot[Sin[x],{x,-Pi,Pi}]

Out[2]=



-Graphics-

In[3]:= Plot[Sin[x],{x,-Pi,Pi}];

In[4]:= Exit

図 8:

ターネットの代名詞ともなっているウェブブラウザをユーザインターフェースとして利用することと、複数の種類の数式処理システムを統一的に扱うことができることである。これらよって、これから数式処理システムを利用しようとするユーザにとっての垣根を低くすることが可能となる。ブラウザ上でどのようなユーザインターフェースが最も使いやすいかということについては研究が必要である。今回はブラウザを利用するユーザがテキスト入力する場合を想定したため、日本語入力からの翻訳を例にとったが、GUI を利用する場合は裏でやりとりされる情報は簡潔な方がよい。数式処理システム一般に対しての統合的な表記方法についても今後の課題としたい。

また、各数式処理システム特有の機能をうまく利用するため、ミドルレギュレータが送られてきた式の特性に応じて処理するという、数式処理システムを自動的に選択する機能についても今後の課題としたい。

参 考 文 献

[1] DEGUCHI Hiroaki: The Integrated Use of Computer Algebra Systems across the Internet, *Proceedings of Asian Symposium on Computer Mathematics*, 1998, pp.101-105

[2] MathLink Reference Guide Version 2.2, Wolfram Research, Inc., 1993

[3] Todd Gayley: A MathLink Tutorial, *http://www.mathsource.com/cgi-bin/MathSource/Enhancements/MathLink/0206-693*

[4] Masayuki Noro and Takeshi Shimoyama: Asir User's Manual, *ftp://endeavor.fujitsu.co.jp/pub/isis/asir/doc/man.ps.gz*